02

Interceptando el Requests

Transcripción

[00:00] ¿Qué tal? Bienvenidos a su quinta clase de su curso de Spring Boot aplicando buenas prácticas y asegurando un API Rest. Hasta el momento lo que hemos visto es el proceso de autenticación, vamos hacer un pequeño resumen sobre eso primero.

[00:15] Usamos Insomnia para simular un cliente front-end, aplicación móvil cualquier cliente y enviamos un usuario, una clave, ya aprendimos cómo obtener estos parámetros, lo enviamos a nuestro API Rest.

[00:28] Y ya como estamos integrando con nuestra base de datos, ya con nuestro método login ya podemos generar nuestros usuarios, la clave siempre tenerla encriptada con el algoritmo BCrypt que es el que hemos elegido. Una vez que Spring encuentra este usuario en la base de datos, la base datos nos lo va a devolver, Spring va hacer la comparación si el usuario es de que hemos requerido o no.

[00:52] Vale la pena recalcar aquí que ese comportamiento es una autenticación del tipo stateless a diferencia de stateful, porque al inicio apenas instalamos Spring Security, ustedes también ya vieron que Spring por defecto ya nos crea una interfaz para hacer un login con un usuario y una clave.

[01:12] Entonces aplicando algunas modificaciones en el código, por ejemplo, implementando algunas interfaces propias de Spring, por ejemplo UserDetails, la interfaz UserDetails en nuestra clase usuario para indicarle que el usuario es un usuario de la autenticación de Spring, y uno una entidad cualquiera.

[01:31] Ya podemos indicarle también los campos, en fin, ustedes ya conocen mejor todas las mejoras y personalizaciones que hemos hecho para conseguir sobrescribir ese comportamiento y pasar de una autenticación stateful a stateless. ¿Qué más hemos visto?

[01:49] Hemos visto cómo generar nuestros propios JWT, JSON Web Token, aprendimos cómo agregar campos en nuestro token, generarnos una librería de Auth0 y finalmente devolver ese token a nuestro cliente.

[02:06] Por ejemplo, vamos a ver aquí en nuestro cliente. Aquí sí ya nos logueamos como “diego.rojas”, con mi clave, login, clave, no hay nada del otro mundo. El nos devuelve ese token. Y ahora la pregunta es ¿qué vamos a hacer con ese token? Por ejemplo, ¿recuerdan que iniciamos la clase anterior diciendo que esta no es una API pública? Porque es de una clínica.

[02:30] Por lo tanto mis recursos, mis recursos de médico todo eso deberían estar protegidos. Pero si yo sigo aquí enviado el request, vemos que siempre está obteniendo los datos, siempre está abierto. Entonces esto es lo que vamos a tratar de cerrar hoy día, vamos a bloquear todos nuestros requests para que acepten requests solo con un token válido.

[02:57] Si recuerdan la presentación anterior, esto ya sería el proceso de autorización. Recuerden el ejemplo que les platiqué. Autenticación es cuando tú válidas que tú eres quien dice ser, esto quiere decir que por ejemplo con tu documento de identidad, tú puedes probar que tú eres la persona que está en ese documento.

[03:20] En nuestro caso, con nuestro usuario y nuestra clave probamos que somos nosotros. Eso es lo que es la autenticación. Y autorización es qué tanto puedes hacer tú en esa aplicación. ¿Y cómo es que funciona esto? Viendo un poco más detalladamente, el token que acabamos de generar aquí en el cliente, tiene que ser enviado en el header o en los encabezados en las cabeceras de request, porque recuerden que request tiene headers y tiene body.

[03:48] Entonces en el header siempre todo request tiene que ir acompañado de un token válido. Recuerden que a nuestro token le hemos configurado un tiempo de expiración. Entonces si ese tipo de expiración aún no caduca, debería ser aceptado en nuestro request, solo en ese caso.

[04:07] Y bueno en el body es la información que enviamos normalmente. Del lado del API Rest, por ejemplo, si recibe un body con authentication, lo primero que va a hacer es validar el token, y si es que está validado liberamos acceso. Si no está validado entonces bloqueamos el request completamente y enviamos a un mensaje de Unauthorized por ejemplo, un 401 y ahí terminaría ese flujo.

[04:34] Viéndolo un poco más detallado técnicamente aquí, en la parte de Spring por ejemplo, si yo ya les mencioné que cada request tiene que venir con un token, y yo necesito interceptar ese token o ese request para poder evaluar, aquí yo podría hacerlo de dos formas, por ejemplo miren aquí. Este es mi lado del código.

[05:01] Y yo podría ponerle algo así como var token = obtenerToken algo así como esto, ahorita estamos escribiendo pseudocódigo, y yo puedo obtener el token y hacer un If (tokenValido) entonces, haz alguna cosa, pero recuerden que esto de aquí tendría que ir repetirse por cada método en mis otros controllers.

[05:40] Por ejemplo el médico controller, yo tendría que obtener el token aquí para retornar los datos del médico, lo mismo para eliminar médico y todos los métodos en todos los controllers. Esto de copiar y pegar código no es escalable y ya lo vimos con nuestro manejo de excepciones. ¿Qué hicimos en ese caso?

[05:58] Bueno, vinimos a crear nuestro paquete de errores en infra y construimos un tratador de errores, entonces pensando en esto quizás ahora ustedes piensen: “Okay, ¿por qué no implementar filtros?” Si ustedes han llevado ya el curso de Java Servlet, quizás ya son familiares con los filtros y

saben que con un filtro pueden interceptar el request antes que llegue a controller y modificar algunos datos, hacer algunas validaciones. Y es el flujo ideal, digamos.

[06:29] Por ejemplo el request va entrando aquí, tienes tus filtros, entran al DispatcherServlet, que es el Servlet de Spring que hace digamos el redirect al controller correcto y bueno, tenemos otros interceptores, que son los handler interceptors, hasta que llega al controller.

[06:47] Esto de aquí es implementación propia de Spring. Esto es lo que sucede dentro de Spring y esto es lo que tú puedes implementar. Entonces puedes implementar al nivel que incluso antes que Spring actúe, tú ya puedes implementar tus propios filtros y modificar el request o hacer la validación hasta que llega a tu controller o quizás ni siquiera llega al controller porque lo rechazas en este nivel.

[07:10] Eso es lo que vamos a ver en esta parte del curso. Es el proceso de autorización de los requests. Nos vemos en el siguiente video.